Performance Bounds for Flow Control Protocols[1]

Rajeev Agrawal, R. L. Cruz, Clayton Okino, and Rajendran Rajan [2]

April 1998

Abstract

In this paper, we discuss a simple conceptual framework for analyzing the flow of data in integrated services networks. The framework allows us to easily model and analyze the behavior of open loop, rate based flow control protocols, as well as closed loop, window based flow control protocols. Central to the framework is the concept of a service curve element, whose departure process is bounded between the *convolution* of the arrival process with a *minimum service curve* and the convolution of the arrival process with a *maximum service curve.* Service curve elements can model links, propagation delays, schedulers, regulators, and window based throttles. The mathematical properties of convolution allow us to easily analyze complex configurations of service curve elements to obtain bounds on end-to-end performance. We demonstrate this by examples, and investigate tradeoffs between buffering requirements, throughput, and delay, for different flow control strategies.

Keywords: Guaranteed service, adaptive service, network calculus, service curve, regulator, scheduler, delay, burstiness, queueing.

[1]Research supported in part by the National Science Foundation under grants NCR-93-05018 and NCR 94-15684, the Army Research Office under grant FRI DAAH04-95-1-0248, the Air Force Office of Scientific Research under grant F49620-95-1-0538, and by the Center for Wireless Communications at UCSD.

[2]Corresponding author. Mailing address: IBM T. J. Watson Research Center, Hawthorne, NY, 10532, e-mail: `raju@watson.ibm.com`

I. Introduction

In this paper, we present and analyze two service categories – *guaranteed service* and *adaptive service* – in integrated services networks. The guaranteed service is designed for applications with stringent delay and loss requirements on a per-packet basis, that are able to specify reasonable upper bounds on their traffic. Typical applications targeted by such a service include audio and video teleconferencing, as well as playback. The adaptive service, on the other hand, is intended for applications such as web browsers that could have a minimum bandwidth requirement, relaxed delay requirements and use window flow control to obtain bandwidth unused by the network. These service categories are in the same spirit as those adopted by the IETF Int-Serv working group [23, 30].

We study provisioning requirements and characterize performance for guaranteed and adaptive sessions. A typical session of either service category traverses a network composed of heterogenous elements. In Section II, we first evolve a simple mathematical framework to capture session dynamics at such diverse elements. The basis of this framework is an operation called *convolution*, through which we define a *service curve element* – a generic device that can equally well be used to describe the operation of a variety of network elements such as links, access regulators, and routers which use link scheduling mechanisms to arbitrate amongst sessions. In this model, *minimum* service curves capture a lower bound on the amount of service allocated by the network element to a given flow, while *maximum* service curves describe a similar upper bound. We conclude Section II by studying the performance characteristics of a flow traversing a single service curve element, and obtain upper and lower bounds on buffering requirements, delay and output burstiness of the flow.

In Section III, we describe the guaranteed service. Briefly this service is as follows. The source describes the offered traffic in terms of a *profile* or *envelope* and requests a lossless service with a fixed upper bound on end-to-end delay. Each of the routers on the path of the session allocates a certain amount of link bandwidth and buffers in order to satisfy this session's request. The routers may reshape the session to its requested profile, and hence the session does not have any assurances on how traffic in excess of the profile will be treated. We show that a unicast guaranteed session may be modelled as a flow traversing a series of service curve elements. Such a series, may be collapsed into a single network element, thereby enabling us to apply the single service element results to obtain end-to-end bounds on queue lengths, delay and output burstiness for such a session, and use these to study the use of regulators in reducing buffering requirements at intermediate network elements. The idea of collapsing the model of a network for a single session to a single network element for purposes of end-to-end analysis was first proposed by Parekh and Gallager [20, 21, 22], in the context of a specific scheduling algorithm.

In Section IV we introduce the adaptive service category. Such a service is useful for applications such as file transfers and web browsing that can use more bandwidth when it is available in the network, while reserving a minimal level of service at all times. In order to avail of excess bandwidth, such sessions require feedback from the network. In this case, the session topologies may be modelled as a *cycle* of service curve elements, with a *throttle* that controls access to the cycle. Our formulation allows us to derive novel pathwise performance bounds for such networks. In particular, we consider unicast as well as multicast sessions, with end-to-end and hop-by-hop window flow control. We show how a *cycle* of service curve elements may be collapsed to a single service curve element. This allows us to obtain the relationship between the buffering requirements at network elements and the performance achievable by the session for the above mentioned session topologies.

II. Modelling and Analysis of Network Elements

In this section, we model a network element – be it a T-1 line, a router, or an access regulator – in terms of the transformation effects the element has on the stream of packets belonging to a session. We present a mathematical model, called a *service curve element*, which specifies how the arriving stream of packets (the arrival process) is converted into a departing stream (the departure process). We show that the service curve element can effectively model a variety of network elements. In the next section, we describe how service curve elements can be concatenated to model network dynamics as experienced by a session.

To this end, we define a *process* to be a function of time $A(t)$. Formally, A is a mapping from the real numbers into the extended non-negative real numbers, i.e. $A : \mathbb{R} \to \mathbb{R}_+ \cup \{+\infty\}$. A process could count the amount of data arriving or departing to/from some network element, and in this case we may call the process an arrival process or a departure process, respectively. All processes are assumed to be non-decreasing and right continuous. We shall often consider what we call *causal* processes, which are simply processes which are identically zero for all negative times. For example, if A is a causal arrival process to a network element, then $A(t)$ is equal to the amount of data (in bits) arriving to the network element in the interval $(-\infty, t]$, and $A(t) = 0$ for all $t < 0$.

A. Service Curve Elements

In order to define service curve elements, we first introduce an operation called *convolution.* Given two processes A and B, the convolution of A and B is defined to be the function $A * B : \mathbb{R} \to \mathbb{R}_+ \cup \{+\infty\}$ such that

$$A * B(t) := \inf_{\tau \in \mathbb{R}} \{A(\tau) + B(t - \tau)\} \ .$$

It is easy to verify that $A*B$ is a process, i.e. it is non-decreasing, and right continuous. Furthermore, if A and B are causal, then $A * B$ is causal. A graphical interpretation of convolution is illustrated in Figure 1 and discussed below.

Suppose that the arrival of traffic to a network element is described by the cumulative arrival process A. Suppose S and $\bar{S}$ are causal processes. The network element is a *service curve element* with *minimum service curve* S (resp. *maximum service curve* $\bar{S}$) if the corresponding departure process D from the element satisfies $D \geq A * S$ (resp. $D \leq A * \bar{S}$)[1]. In the next subsection, we show that these models allow us to describe the dynamics of a variety of network elements, including for instance, a fixed propagation delay, an access regulator or a queueing server with a link scheduling mechanism.

In order to make sense of the definition of service curve elements, it is important to visualize the concept of convolution. To this end, we now discuss a graphical interpretation. For a fixed value of τ, the graph of $A(\tau) + B(t - \tau)$ versus t is obtained from the graph of $B(t)$ by horizontally shifting it by an amount τ and vertically shifting it by an amount $A(\tau)$. In other words, we translate the graph of $B(t)$ by moving the origin of the graph onto the point $(\tau, A(\tau))$. By taking the pointwise minimum of all such translations of the graph of B onto the graph of A, we obtain the convolution.

[1]In this paper, all inequalities involving functions are defined in a pointwise sense.

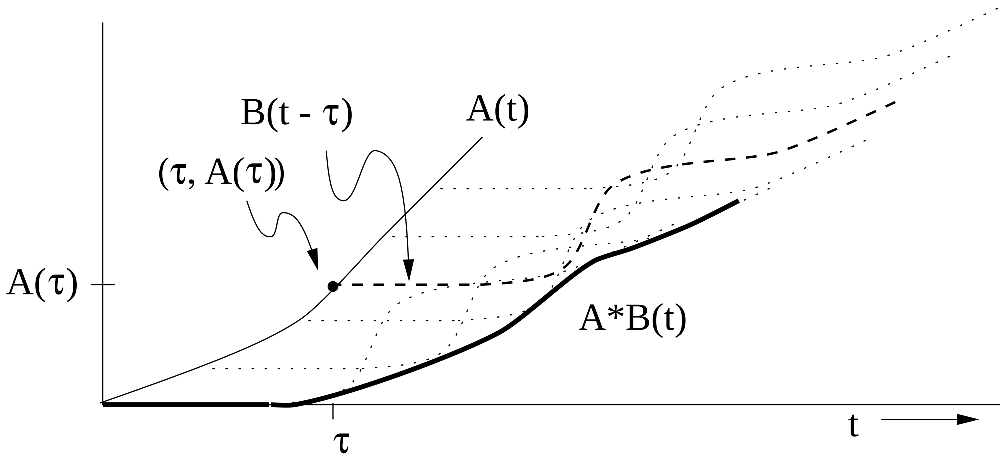

Fig. 1. Graphical representation of convolution $A * B$.

This is illustrated graphically in Figure 1.

With this graphical interpretation, it can be seen (and easily verified formally) that the convolution operator is **commutative** and **associative** , i.e. $A * B = B * A$ and $(A * B) * C = A * (B * C)$. Furthermore, if $B \wedge C$ denotes the pointwise minimum of B and C, then $A * (B \wedge C) = (A * B) \wedge (B * C)$. In other words, convolution is **distributive** with respect to the minimum operator. The identity element δ of this operator that satisfies $A * \delta = \delta * A = A$ may be verified to be

$$\delta(t) := \begin{cases} 0 & t < 0 \\ \infty & t \geq 0 \ . \end{cases}$$

Defining $\delta_d(t) := \delta(t - d)$, note that δ_d is a "shift element," i.e. for any process A we have $A * \delta_d(t) = A(t - d)$. If B is a causal process then $B \leq \delta$ and hence $A * B \leq A * \delta \leq A$ for any process A.

B. Modelling Network Elements as Service Curve Elements

In this subsection, we describe four types of *service curve elements* - delay element, link, scheduler, and regulator. Using combinations of the four service curve elements, a variety of network elements may be modelled for a guaranteed service session, as we demonstrate in Example 2 at the end of this subsection.

1. Delay Elements

Consider a traffic stream passing through a network element, where A and D denote the arrival and departure processes, respectively. Suppose that the delay is bounded above by d_{max} and below by d_{min} in the sense that

$$A(t - d_{max}) \leq D(t) \leq A(t - d_{min}) \quad \text{for all } t \in \mathbb{R}.$$

Recalling the shift element δ_d defined earlier, this is equivalent to

$$A * \delta_{d_{max}} \leq D \leq A * \delta_{d_{min}} \ .$$

Thus, the above network element is a service curve element with minimum service curve $\delta_{d_{max}}$ and maximum service curve $\delta_{d_{min}}$. Note that any physically realizable network element is a service curve element with a maximum service curve of δ, since $D \leq A = A * \delta$. For a constant propagation delay element the minimum and maximum service curves are identical, namely $\delta_{d_{prop}}$, where d_{prop} is the value of the propagation delay.

2. Links

Suppose a link has a capacity of C bits per second, in the sense that $D(t) - D(\tau) \leq C(t - \tau)$ for all $\tau \leq t$. Since $D \leq A$, this implies that $D(t) \leq A(\tau) + C(t - \tau)$, and hence that $D \leq A * R_C$, where $R_C(t) = Ct$ for $t \geq 0$ and $R_C(t) = 0$ for $t < 0$. Thus, a network element with a capacity of C bits per second is a maximum service curve element with maximum service curve R_C. If the session under consideration is the only session on this link then the above inequality holds with equality and the network element guarantees both a maximum and minimum service curve of R_C. Recognize that this is equivalent to the behavior of a fixed rate server.

In general, given a network element that guarantees a maximum service curve of $\bar{S}$, note that the maximum throughput is upper bounded by the asymptotic slope of $\bar{S}$, i.e.

$$\begin{aligned} \overline{\lim_{t\to\infty}} \frac{D(t)}{t} &\leq \overline{\lim_{t\to\infty}} \frac{A * \bar{S}(t)}{t} \\ &\leq \overline{\lim_{t\to\infty}} \frac{A(0) + \bar{S}(t)}{t} \\ &= \overline{\lim_{t\to\infty}} \frac{\bar{S}(t)}{t} \ . \end{aligned}$$

3. Schedulers

Minimum service curves may also be used to characterize the level of service provided by a link scheduler to each of the sessions sharing the link. Such service characterization may reflect active scheduler involvement in isolating flows (as in virtual clock, generalized processor sharing (GPS), packet GPS, self-clocked fair queueing, weighted round-robin, etc.) or may be obtained through a careful accounting of the service capacity and arrival constraints on other connections (as when several burstiness constrained flows share a FIFO or priority scheduler). Service curves, with a somewhat different definition than in this paper, were introduced by Parekh and Gallager [21, 22] to study generalized processor sharing (GPS). The idea of using a service curve as a general characterization of a scheduling policy was proposed by Cruz [7], and refined in [8]. Closely related service definitions, which are a special case of the service curve framework in this paper, were made by Stiliades and Varma [26], Hung and Kesidis [16], and Goyal, Lam, and Vin [15]. These three latter definitions are essentially equivalent, and have recently been considered by the integrated services working group of the IETF. We call them "latency rate" service curves, following the terminology of Stiliades and Varma. It should be noted that the minimum service curve definition of this paper was concurrently proposed by Agrawal and Rajan [1], Le Boudec [18], and reported in the thesis of Sariowan [25].

We now discuss the latency rate service model adopted by the integrated services working group of the IETF. Consider the series combination of a fixed rate server with capacity μ bits per second and a constant delay element with a delay of d seconds. If the arrival process is A, then it follows that the corresponding departure process from the fixed rate server is $A * R_\mu$, and hence the corresponding departure process from the delay element is $(A * R_\mu) * \delta_d = A * (R_\mu * \delta_d)$. Thus, if a network element serves data at least as fast as this series combination, it guarantees a minimum service curve of $R_\mu * \delta_d$. The parameter d is called the latency and the parameter μ is called the rate. Latency rate service curves bound the dynamics of a number of popular scheduling mechanisms – self-clocked fair queueing [14], worst-case-fair weighted-fair queueing [2], and virtual clock [33], for example.

In general, given a minimum service curve S, it is possible to *synthesize* a scheduling algorithm so that a server guarantees the service curve S to a given traffic stream. This philosophy is taken in [25] [24] [27], and is closely related to the "Earliest Deadline First" (EDF) scheduling policies considered in [11] and [19].

4. Regulators

Regulators are devices used to shape or smooth traffic to an envelope, and may be used to model devices such as leaky buckets that enforce traffic contracts at access points to the network or subnetworks. Further, regulators may also be used within a network, for instance, when link scheduling is performed using a *rate controlled service discipline* [31, 32] which uses regulators to isolate incoming traffic streams before arbitrating between competing streams through a scheduling mechanism. This use of regulators has the additional advantage of reducing jitter and buffering requirements at downstream network elements [6, 12], and is related to the stop & go scheduling policy proposed by Golestani [13].

In order to formally define a regulator, it is first neccessary to introduce the notion of an *envelope*. A process E is said to be an *envelope* for the process A if for all $\tau \le t$ we have $A(t) - A(\tau) \le E(t - \tau)$, or equivalently $A \le A * E$. The concept of an envelope was proposed and developed in [5].

If E is an envelope for A then $E \wedge \delta$ is a causal envelope for A, since any process A is non-decreasing[2]. If E is a causal envelope, note that $E \le \delta$, and hence $A * E \le A * \delta = A$. Thus, if E is a causal envelope for A, then $A = A * E$.

A process E is said to be *sub-additive* if for all $t, \tau \in \mathbb{R}$ we have $E(\tau) + E(t - \tau) \ge E(t)$. Thus, if E is a sub-additive process, then $E * E \ge E$. We will often assume that envelopes are subadditive, and given the definition of an envelope, this is a natural assumption [3]. A simple example of an envelope is the (σ, ρ) envelope with $E(t) := \sigma + \rho t, t \ge 0$.

The (σ, ρ) regulator is a device that shapes traffic to a (σ, ρ) envelope, i.e., it holds up arrivals just long enough to ensure that the departing stream satisfies $D \le D * E$, where E is a (σ, ρ) envelope. A popular implementation of a (σ, ρ) regulator is called a *leaky bucket* [29], where tokens arrive into a bucket of size σ at constant rate ρ. Departures occur only when tokens are available, and when they occur, cause the contents of the token bucket to be decremented by an equal volume. The (σ, ρ) regulator was introduced by Cruz [5, 8] and further generalized by Anantharam and

[2]It will be occasionally convenient to consider non-causal envelopes.

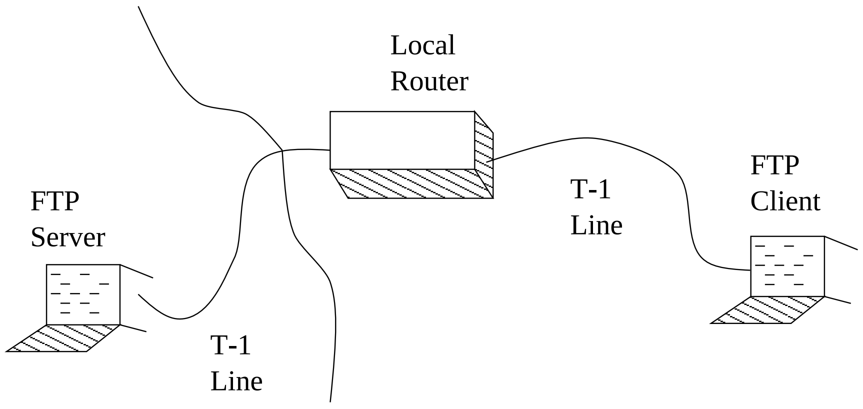

Fig. 2. A Single Hop File Transfer Session.

Konstantopoulos [17].

The departure process D of a regulator with causal sub-additive envelope E and arrival process A satisfies the following conditions:

R1. E is an envelope for D, i.e. $D \le D * E$.

R2. $D \le A$.

R3. D is the (pointwise) maximal function satisfying **R1** and **R2**.

The departure process satisfying the above conditions **R1**–**R3** is explicitly obtained in the following theorem.

Theorem 1: There exists a unique D satisfying **R1**–**R3** and is given by $D = A * E$.

Proof: From the sub-additivity of E it follows that $D' := A * E \le A * (E * E) = (A * E) * E = D' * E$. Thus, E is an envelope for D'. Since E is causal, we have $E \le \delta$, and hence $D' := A * E \le A * \delta = A$. Thus, D' satisfies **R1** and **R2**. Let D'' be any departure process that also satisfies **R1** and **R2**. Then $D'' \le D'' * E \le A * E = D'$, and consequently, D' satisfies **R3**. $\square$

Thus the envelope of a regulator is both its minimum and maximum service curve. Essentially the same result has been concurrently reported in [1], [4], and [25].

Example 2 (Modelling Networks with Service Curve Elements): Consider the example of a Guaranteed Service session from a source to a destination across one router depicted in Figure 2. Suppose that the session has a traffic profile with burst size 5 Kb, and a token rate of 200 Kbps. The session

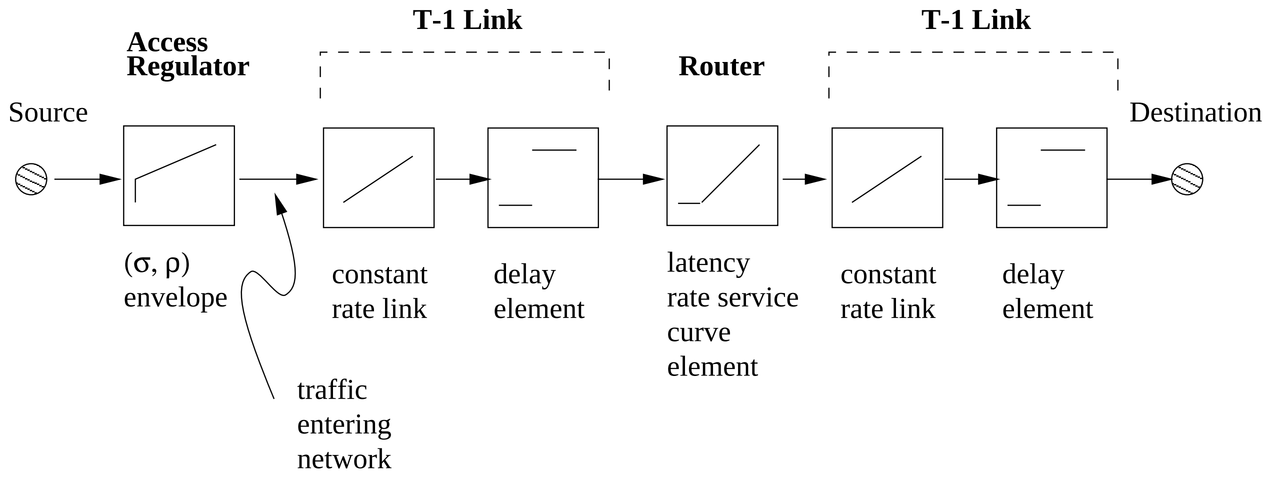

Fig. 3. Service Element Model for FTP session.

traverses a 1.5 Mbps T-1 link with a propagation delay of 3 ms before being forwarded on to the second link by the router. We assume that the scheduling policy in the router can be modelled as a latency-rate server, with a latency of 1 ms and rates of 300 Kbps[3]. The second T-1 link has a propagation delay of 2 ms. We may model this session as a series of service curve elements as shown in Figure 3. In Figure 3, the source traffic is modelled as emanating from an access regulator, which has a (σ, ρ) envelope with parameters $\sigma = 5$ Kb, $\rho = 200$ Kbps. The first T-1 line is modelled as a delay element of 3 ms followed by a constant rate server of 1.5 Mbps, while the second is a delay element of 2 ms followed by the same constant rate server. Finally, the operation of the router is modelled by a latency rate (scheduler) service curve with latency 1 ms and a rate of 300 Kbps. We defer performance calculations until later in the paper (See Example 12).

C. Performance bounds for single service curve elements

In this subsection, we derive delay bounds, buffer requirements and departure process envelope for a session traversing a single service curve element. To aid us in proving these bounds we first describe an operation called *deconvolution.*

1. Deconvolution

Given an envelope E, it is common to consider the class of all processes for which E is an envelope. Conversely, given a causal process A, it is also useful to consider the smallest envelope E for A.

[3]The larger server rates assure a smaller end-to-end delay for the session.

This is clearly given by

$$E(t) = \sup_{\tau \in \mathbb{R}} \{A(t+\tau) - A(\tau)\} \ .$$

Note that E above is the smallest process such that $A(t+\tau) - A(\tau) \leq E(t)$ for all $\tau, t \in \mathbb{R}$. More generally, given two processes A and B, where B is causal, it is useful to consider the smallest process H such that $A(t+\tau) - B(\tau) \leq H(t)$ for all $t, \tau \in \mathbb{R}$. The smallest such process is clearly given by $H = A \oslash B$, where[4]

$$A \oslash B(t) = \sup_{\tau \in \mathbb{R}} \{A(t+\tau) - B(\tau)\}.$$

Note that the condition above that $A(t+\tau) - B(\tau) \leq H(t)$ for all $t, \tau \in \mathbb{R}$ is equivalent to the condition that $A(t) - B(\tau) \leq H(t-\tau)$ for all $t, \tau \in \mathbb{R}$. In turn, it is easily seen that this condition is equivalent to $H * B \geq A$. Thus, $A \oslash B$ is the smallest process H such that $H * B \geq A$. For this reason, we call $A \oslash B$ the *deconvolution* of A and B, or A deconvolved with B.

It can easily be verified that $(A \oslash B) \oslash C = A \oslash (B * C)$ for any processes A, B, and C. Furthermore, the following lemmas, proven in the Appendix, will be useful later.

Lemma 3: For any processes A,B,C, where C is causal, there holds $A * (B \oslash C) \geq (A * B) \oslash C$.

Proof: See Appendix. $\square$

Lemma 4: For any causal process B, the process $B \oslash B$ is sub-additive.

Proof: See Appendix. $\square$

A consequence of the last lemma is that the smallest envelope E of a given process A, $A \oslash A$, is sub-additive. This fact was first observed by Chang [3].

2. *Delay Bounds, Buffer Requirements, and Departure Traffic Characterization*

Throughout this subsection we will consider a service curve element with arrival process A and departure process D. We will assume that E is an envelope for the arrival process A, and that the minimum and maximum service curves are S and $\bar{S}$, respectively. Most of the results of this

[4]It is easy to verify that $A \oslash B$ is a process, i.e. it is non-decreasing, right continuous, and non-negative since B is causal.

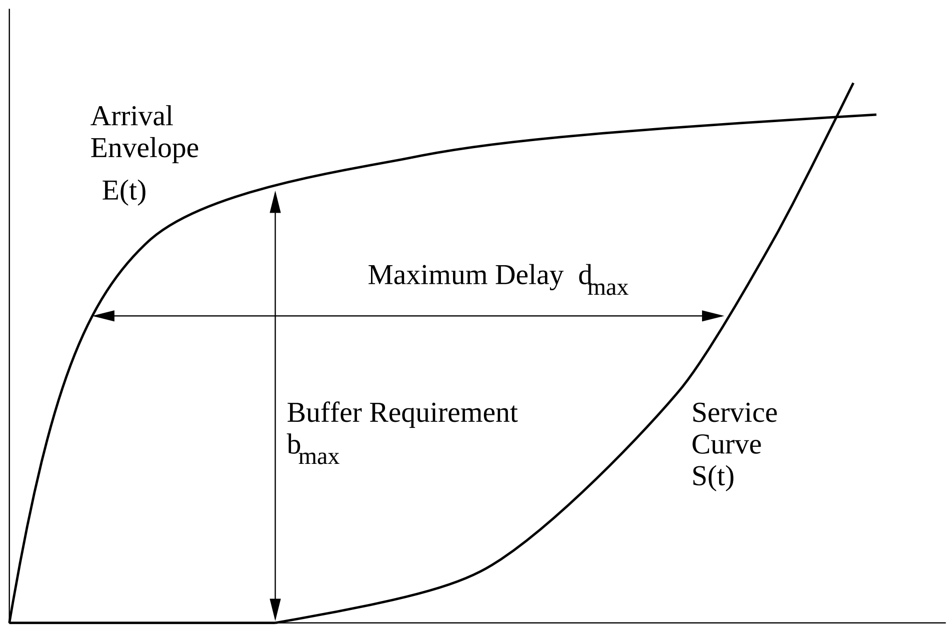

Fig. 4. The worst case delay and buffer requirements.

subsection appeared in [1], [18], [25], and in an earlier form in [8]. The proofs here are considerably more streamlined, and are adapted from [9].

Under these conditions, we first obtain bounds on the delay through the network element. Let d_{max} be the maximum horizontal distance between E and S. In other words, d_{max} is how far the graph of E must be shifted to the right so that it lies below S:

$$d_{max} := \inf\{d : d \geq 0 \ , \ E * \delta_d \leq S\} \tag{1}$$

It is easy to verify that $E * \delta_{d_{max}} \leq S$. Figure 4 is an example illustrating d_{max}.

Proposition 5: The service curve element has a delay of at most d_{max}, i.e. $D(t) \geq A(t - d_{max})$ for all t.

Proof: Using the definition of a minimum service curve, the fact that $E * \delta_{d_{max}} \leq S$, the definition of d_{max}, associativity, and the definition of a traffic envelope, we have

$$\begin{aligned} D &\geq A * S \\ &\geq A * (E * \delta_{d_{max}}) \\ &= (A * E) * \delta_{d_{max}} \\ &\geq A * \delta_{d_{max}} \ . \end{aligned}$$

$\square$

A lower bound on the delay can be easily obtained from the maximum service curve. Towards this end, let $d_{\min}$ be the largest value of t such that $\bar{S}(t) \leq 0$, i.e. $d_{\min} = \sup\{t : \bar{S}(t) = 0\}$. Thus, we have $\bar{S}(t) = 0$ for $t < d_{\min}$ and hence $\delta_{d_{\min}} \geq \bar{S}$.

Proposition 6: The service curve element has a delay of at least $d_{\min}$, i.e. $D(t) \leq A(t - d_{\min})$ for all t.

Proof: Using the definition of a maximum service curve and the definition of $d_{\min}$ we have $D \leq A * \bar{S} \leq A * \delta_{d_{\min}}$. $\square$

Next, we bound buffer requirements. Towards this end, let b_{max} be the maximum vertical distance between E and S, i.e.

$$b_{max} := \sup\{E(t) - S(t) : t \in \mathbb{R}\} = E \oslash S(0) \ . \tag{2}$$

An example of b_{max} is illustrated in Figure 4.

Proposition 7: The amount of data stored at any time in the network element is at most b_{max}, i.e. $A(t) - D(t) \leq b_{max}$ for all t.

Proof: Note that $E \leq b_{max} + S$. Using the definition of a traffic envelope and a minimum service curve, we thus have

$$\begin{aligned} A(t) - D(t) &\leq A * E(t) - A * S(t) \\ &\leq A * (b_{max} + S)(t) - A * S(t) \\ &= b_{max} + A * S(t) - A * S(t) \\ &= b_{max} \ . \end{aligned}$$

$\square$

Next, we find a traffic envelope for the departure process D in terms of the traffic envelope for the arrival process and the maximum service curve $\bar{S}$.

Proposition 8: The process $E_{out} := (E * \bar{S}) \oslash S$ is an envelope for the departure process[5].

[5]Note that E_{out} is not necessarily causal. Of course $E'_{out} := E_{out} \wedge \delta$ is a causal envelope for the departure process.

Proof: It suffices to show that $D \leq D * E_{out}$. We have

$$\begin{aligned} D * E_{out} &= D * ((E * \bar{S}) \oslash S) \\ &\geq (A * S) * ((E * \bar{S}) \oslash S) \\ &= A * (((E * \bar{S}) \oslash S) * S) \\ &\geq A * (E * \bar{S}) \\ &= (A * E) * \bar{S} \\ &\geq A * \bar{S} \\ &\geq D\ . \end{aligned}$$

□

In case the network element in question is a regulator, we obtain the following corollary. In a discrete time context, this result was previously obtained by Chang [4].

Corollary 9: Suppose E_{in} is an envelope for the arrival process to a regulator with causal envelope E_{reg}. Then E_{out} is an envelope for the departure process, where $E_{out} = E_{in} * E_{reg}$.

Proof: Applying Proposition 8 and observing that the envelope E_{reg} is the minimum and maximum service curve of the regulator, it follows that the departure process has envelope $(E_{in} * E_{reg}) \oslash E_{reg}$. On the other hand, using Lemma 3 and the sub-additivity of E_{reg} we have $(E_{in} * E_{reg}) \oslash E_{reg} \leq E_{in} * (E_{reg} \oslash E_{reg}) \leq E_{in} * E_{reg}$. □

III. Guaranteed Service Sessions

A session belonging to the Guaranteed Service category specifies the offered load in terms of an envelope (a (σ, ρ) envelope in the simplest instance), and requests a certain level of service, quantified by a worst-case end-to-end delay requirement, and no packet loss. For the call to be setup successfully, network resources are reserved to ensure the requested level of performance. We model such a session as a feed-forward network of service curve elements. Thus, a single element may represent a fixed propagation delay, a queueing server, or a router. The principal advantage of the service curve framework in analyzing such complex configurations is that we can collapse the network into a single service curve element. This allows us to apply the bounds obtained for a single element in the previous section, to analyze complex networks.

A. Composition Rule for Tandem Service Curve Elements

Consider a feedforward series of network elements, whereby data flows sequentially through a set of network elements. In this case, a service curve of the system is obtained by convolving the service curves of each of the network elements. This is formalized in the following :

Proposition 10: Suppose a traffic stream passes through n service curve elements in series, where the i^{th} element has minimum service curve S_i and maximum service curve $\bar{S}_i$, $i = 1, 2, \ldots n$. Then the entire system is a service curve element with minimum and maximum service curves $S_1 * S_2 * \cdots * S_n$ and $\bar{S}_1 * \bar{S}_2 * \cdots * \bar{S}_n$, respectively.

Proof: We prove the proposition for the case $n = 2$. The general case can then be proved by induction on n. Let the arrival process to the system be A, and the departure process of the first element be B. Note that B is also the arrival process to the second element. Finally, let D be the departure process of the second element. We have

$$\begin{aligned} D &\geq B * S_2 \\ &\geq (A * S_1) * S_2 \\ &= A * (S_1 * S_2) \ . \end{aligned}$$

Thus, the system guarantees a minimum service curve of $S_1 * S_2$. Similarly we have

$$\begin{aligned} D &\leq B * \bar{S}_2 \\ &\leq (A * \bar{S}_1) * \bar{S}_2 \\ &= A * (\bar{S}_1 * \bar{S}_2) \ . \end{aligned}$$

and thus the system guarantees a maximum service curve of $\bar{S}_1 * \bar{S}_2$. □

Combining Propositions 5–10, we may obtain end-to-end performance bounds as illustrated in the following examples.

Example 11 (Tandem of latency-rate service curve elements fed by a sigma-rho process): Consider an arrival process constrained by a (σ, ρ) envelope that feeds a tandem of n service curve elements. Element k has a latency-rate (θ_k, r_k) minimum service curve and latency rate (ϕ_k, p_k) maximum service curve, $1 \leq k \leq n$. Assume that $\phi_k \leq \theta_k$ and $\rho \leq r_k \leq p_k$. Then, it is easy to verify that the system guarantees a minimum service curve which is of the latency rate type with latency $\theta = \sum_{k=1}^{n} \theta_k$ and rate $r = \min_{1 \leq k \leq n} r_k$ and also a maximum service curve which is of the latency rate type with latency $\phi = \sum_{k=1}^{n} \phi_k$ and rate $p = \min_{1 \leq k \leq n} p_k$. In this case the minimum end-to-end delay is θ, the maximum end-to-end delay is $\theta + \sigma/r$, the end-to-end queue length bound is $\sigma + \rho\theta$, and the envelope of the departure process from the system is given by

$$E_{out}(t) = p\frac{\sigma}{p-\rho} + (\rho(t - \frac{\sigma}{p-\rho} + \theta - \phi)) \wedge (r(t - \frac{\sigma}{p-\rho} + \theta - \phi)), \quad t \geq 0.$$

Example 12 (Performance for Example 2): In Figure 3, the Guaranteed Service session is depicted as a (σ, ρ) regulator followed by five service curve elements. Specifically, we have envelope $E(t) = 5\text{Kb} + 200\text{Kb/s} \cdot t$, constant rate link $S_1(t) = 1.5\text{Mb/s} \cdot t$, delay element $S_2(t) = \delta_{3\text{ms}}(t)$, latency rate minimum service curve $S_3(t) = \max\{0, -.3\text{Kb} + 300\text{Kb/s} \cdot t\}$, constant rate link $S_4(t) = 1.5\text{Mb/s} \cdot t$, and delay element $S_5(t) = \delta_{2\text{ms}}(t)$. This is a special case of Example 11. We find the net maximum service curve $\bar{S}^{net}(t) = S_1 * S_2 * S_4 * S_5(t) = \max\{0, -7.5\text{Kb} + 1.5\text{Mb/s} \cdot t\}$, and the net minimum service curve $S^{net}(t) = S_1 * S_2 * S_3 * S_4 * S_5(t) = \max\{0, -1.8\text{Kb} + 300\text{Kb/s} \cdot t\}$, and then the minimum end-to-end delay is $d_{min} = \sup\{t : \bar{S}^{net} = 0\} = 5\text{ms}$, the maximum end-to-end delay $d_{max} = \inf\{d : E * \delta_d \leq S^{net}\} = 6\text{ms} + \frac{5\text{ Kb}}{300\text{ Kb/s}} = 22.67\text{ms}$, the end-to-end minimum buffer requirement $b_{max} = E \oslash S^{net}(0) = 5\text{Kb} + 200\text{Kb/s} \cdot 6\text{ms} = 6.2\text{Kb}$, and the departure envelope process $E_{out}(t) = \{4.92\text{Kb} + 300\text{Kb/s} \cdot (t)\} \wedge \{5.2\text{Kb} + 200\text{Kb/s} \cdot (t)\}, t \geq 0$.

B. Reducing Buffering and Service Curve Requirements for Guaranteed Sessions

In this section, we investigate how in a Guaranteed Service session, we may reduce the (i) buffering requirements at schedulers by the insertion of appropriately chosen regulators, and (ii) the minimum service curve requirement at the individual schedulers, while preserving the end-to-end minimum service curve.

We begin with considering regulation at the source.

Example 13 ("Minimum" Network Buffering): Consider a Guaranteed Service session traversing a network which guarantees a minimum service curve of S. In order to limit the amount of data in the network, the source could first pass the data through a regulator with envelope E, so that the amount of data in the network would be no more than $b_{max} = (E \oslash S)(0)$ by Proposition 7. In order to make b_{max} as small as possible, the envelope E should be chosen as small as possible. On the other hand, the system, including the regulator, would then have a minimum service curve of $E * S$. If we desire that the service curve be unchanged with the addition of the regulator, we need $E * S \geq S$. The minimum such envelope E is $E = S \oslash S$ and

$$(S \oslash S) * S = S. \tag{3}$$

The corresponding bound on the amount of data in the network is

$$\begin{aligned} b_{max} &= [(S \oslash S) \oslash S](0) \\ &= [S \oslash (S * S)](0) \\ &= \sup_{t \in \mathbb{R}} \{S(t) - S * S(t)\} \ . \end{aligned} \tag{4}$$

We now motivate the use of regulation inside the network.

Example 14 (Buffer Requirements for Network Elements in Tandem): Consider a Guaranteed Service session that traverses a series of n service curve elements, where the i^{th} element has minimum services curve S_i. Suppose the arrival process to the first network element has envelope E. By Proposition 7, the buffer requirement in the first network element is $b_{max,1} = (E \oslash S_1)(0)$. By Proposition 8, the arrival process to the i^{th} network element has envelope $E \oslash (S_1 * S_2 * \ldots S_{i-1})$ for $i > 1$. Hence by Proposition 7, the buffer requirement in the i^{th} network element is $b_{max,i} = [(E \oslash (S_1 * S_2 * \ldots S_{i-1})) \oslash S_i](0) = [E \oslash (S_1 * S_2 * \ldots S_i)](0)$, which is increasing with i. By lumping the n network elements together and modelling them as a system with minumum service curve S, the buffer requirement for the entire system is $b_{max} = [E \oslash S](0)$, where $S = S_1 * S_2 * \ldots S_n$. Note that the buffer requirement for the system is equal to the buffer requirement for the n^{th} network element, i.e. $b_{max} = b_{max,n}$. Note further that we may have $\sum_{i=1}^{n} b_{max,i} > b_{max}$, even though all the bounds are achievable. The explanation is that the backlogs in the network elements cannot be maximized simultaneously at the same point in time.

It is well known that the buffer requirements can be reduced by the insertion of appropriately chosen regulators between the network elements without affecting the end-to-end maximum delay. The next proposition, proved in the appendix, gives the smallest envelope that can be inserted without affecting the end-to-end minimum service curve, and hence without affecting the maximum end-to-end delay.

Proposition 15: Consider a Guaranteed Service session with source traffic constrained by a causal envelope B, that traverses a tandem of n service curve elements, where the k^{th} element guarantees a minimum service curve S_k. Let $S := S_1 * S_2 * \ldots * S_n$. Let d_{max} be the maximum end-to-end delay encountered by the session in traversing the series of elements. Define the sub-additive envelope $E := (B * S) \oslash (B * S)$. If a regulator with envelope E is inserted before any of the network elements, the end-to-end minimum service curve is unchanged, and hence the maximum end-to-end delay does not increase. By the same argument reducing each of the minimum service curves S_k to $S_k * ((B * S) \oslash (B * S))$ does not change the end-to-end delay.

The next example gives an application of Proposition 15.

Example 16: For a tandem of n identical latency-rate (θ, r) schedulers fed by a (σ, ρ) constrained arrival process considered in Example 11 (with $\rho \leq r$), the corresponding envelope E in Proposition 15 is given by $E(t) = \min\{rt, \sigma + \rho t\}, t \geq 0$. The use of regulators with this burstiness constraint give a buffering requirement of $E(2\theta) = \min\{2r\theta, \sigma + 2\rho\theta\}$ at element $n \geq 2$. If we used regulators with burstiness constraint $B(t) = \sigma + \rho t$ of the arrival process, then the corresponding buffering requirement at node m would be $\sigma + 2\rho\theta$. By the previous proposition, the use of any of either of these regulators leaves the end-to-end delay unaltered. Both of these buffering requirements are smaller than the original buffering requirement of $\sigma + m\rho\theta$ at node m, $m \geq 2$. More importantly, they do not increase with the number of elements traversed by the session.

In this section, we saw how service curves could be used to model, analyze and provision Guaranteed Service sessions that provide a lossless service with bounds on the worst case end to end delay. Further, we studied the use of regulators within the network in reducing buffering requirements within the network. In the next section, we study another approach for limiting buffer requirements, namely window flow control, which allows the session to make better use of network resources when these are idle.

IV. Adaptive Sessions

The Guaranteed Service category is appropriate for applications which require a pre-specified level of service for a volume of traffic that can be well characterized by an envelope. However, a number of applications such as file transfers and web browsing can use more bandwidth when it is available in the network, while requiring a minimal level of service at all times. In order to avail of excess bandwidth, the sessions need feedback and should not be unnecessarily limited by regulation within the network. In this section, we study a service category tailored to such applications called adaptive service, wherein network resources are reserved to obtain a minimum bandwidth guarantee and prevent buffer overflow, and window flow control is used to avail of excess bandwidth in the network. We discuss the service in the context of a unicast session, model the dynamics of the session using service curve elements, and obtain the relationship between window sizes, resource reservation and performance for this session. We shall consider both *end-to-end* window flow control as well as *hop-by-hop* window flow control. Finally, we end this section with a discussion of *multicast* sessions with end-to-end window flow control.

A. Unicast Sessions with end-to-end windows

Figure 5 illustrates a unicast session that originates at a source and traverses multiple routers before terminating at the destination host. In order for the source to obtain a pre-negotiated level of service, each router must allocate buffers to prevent packet loss, as well as guarantee access to link bandwidth through the operation of a scheduling mechanism. The destination returns acknowledgments for each packet received, and the source uses a window of size W to control the number of unacknowledged packets.

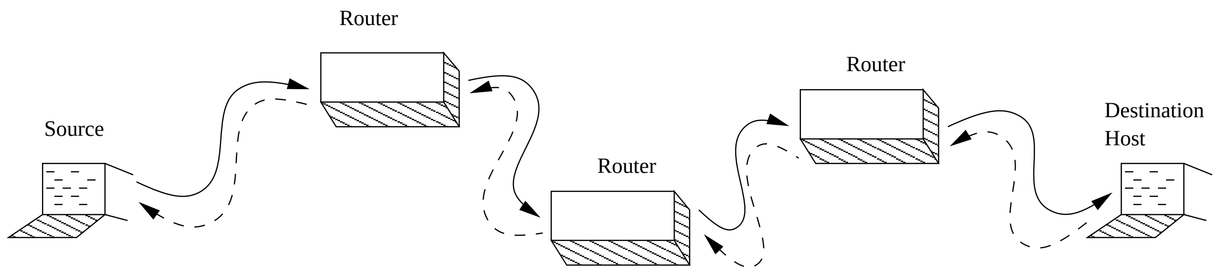

Fig. 5. A unicast session with window flow control.

We model this network using service curve elements, as illustrated in Figure 6. Window flow control is modelled through a network element called a *throttle*, which is controlled by a *throttle process*. In particular, a throttle $\mathcal{T}$ receives traffic from an arrival process, say $A_{\mathcal{T}}$, and releases this traffic according to the departure process $D_{\mathcal{T}} = A_{\mathcal{T}} \wedge T$, where T is the throttle process that controls the throttle. Thus the departure process of a throttle is constrained so that it is never more than the throttle process, and the throttle buffers data, only as necessary, to meet this constraint.

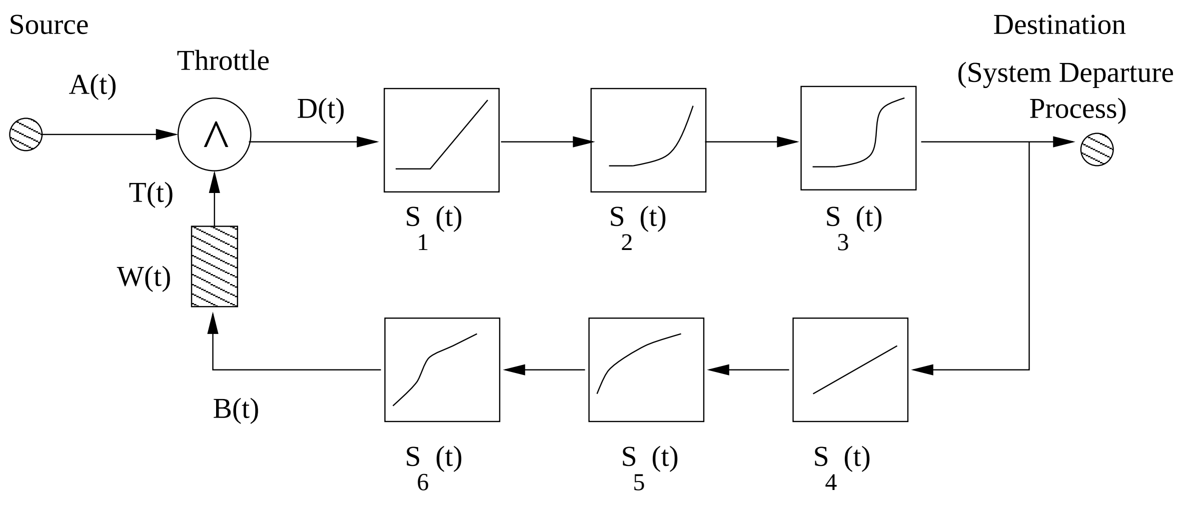

Fig. 6. Service curve elements representing a unicast session with window flow control.

In the model illustrated in Figure 6, the arrival process A to the system is fed to a throttle, and the departure process of the throttle, D, is fed to a tandem of six service curve elements. The departure process of this tandem, B, is summed with another process W, called the *window* process[6], to obtain the throttle process $T = B + W$ that controls the throttle. The amount of traffic stored in the tandem is $(A \wedge T) - B \leq T - B = W$, so that the throttle insures that the amount of traffic in the tandem is no more than W. Thus, the buffering requirement at each of the elements in the tandem is at most $W(t)$ at time t.

The window size can be dynamically adjusted according to any criterion, with the idea that, in order to utilize bandwidth and buffers efficiently it is natural to decrease the window size when the network is congested and to increase it when the network has spare resources. Thus, we may use a TCP-type additive increase and multiplicative decrease algorithm. Suter et al. [28] consider TCP with different scheduling and buffer management schemes to improve the fairness of TCP. The framework is similar to the extent that we consider a combination of link scheduling and dynamic windows. However, it differs in that we consider a lossless model, whereas TCP uses losses

[6]Strictly speaking, W is not a process since we do not require it to be non-decreasing.

explicitly to measure congestion and adjust the window size. In our case we can use some other method to estimate available network resources, while avoiding the possibility of buffer overflow[7]. For instance, we could use round-trip delays, perhaps in combination with some explicit feedback from the network. Our framework also differs in that we may allow the scheduling parameters and the window adjustment mechanisms to differ across sessions, resulting in different pre-defined grades of service.

The issue of how the window process is generated is beyond the scope of this paper. We shall only assume that $B + W = T$ is a process, i.e. non-decreasing, and that the window process is bounded below by a positive constant w^{min} and bounded above by w^{max}:

$$w^{min} \leq W(t) \leq w^{max} \ . \tag{5}$$

Each of the service curve elements in the tandem could represent a link with a fixed propagation delay and maximum bit rate, or a router which provides a certain level of link bandwidth to the session. In Figure 6, the output of the third element in the tandem is the departure process of the system, which is fed to the destination. This departure process is also fed back toward the throttle, and serves as a model of *acknowledgments* for purposes of governing the throttle. The three service curve elements on the return path to the throttle serve as a model for delays (e.g. propagation delays) in the return of acknowledgments to the throttle.

It is important to note that this is merely a model for acknowledgments, and that the data stream itself does not need to return to the throttle. Only the *values* of the departure process are fed back toward the throttle, which requires considerably less bandwidth than the data stream itself. Here, for simplicity, we have assumed that acknowledgments are in the same volume as data traffic. A more realistic assumption is to have the volume of acknowledgments be a fixed fraction of the volume of forward traffic. It is straightforward to extend our analysis to this system.

Given the minimum and maximum service curves of the service curve elements in the tandem, our goal is to show that the overall system is a service curve element, and find minimum and maximum service curves for the system. Toward this end, we first find minimum and maximum

[7]This interesting topic is beyond the scope of the current paper, but is left for future work

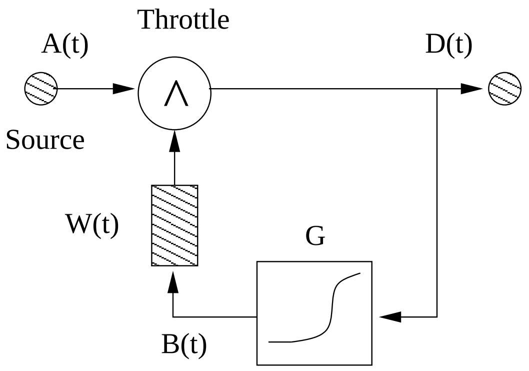

Fig. 7. Basic Model of a Closed Cycle of Service Curve Elements.

service curves of the throttle. In other words, we consider the system, called a *closed cycle* of service curve elements, whose arrival process is A and whose departure process is D, the departure process of the throttle. Once we find a minimum and maximum service curve for the closed cycle, we may apply Proposition 10 to find minimum and maximum service curves of the overall system which feeds the destination, by convolving the service curves of the service curve elements in the forward path of the tandem.

By applying Proposition 10, we can collapse the service curve elements in the tandem into a single service curve element with minimum and maximum service curve G and $\bar{G}$, where $G = S_1 * S_2 * \cdots * S_6$ and $\bar{G} = \bar{S}_1 * \bar{S}_2 * \cdots * \bar{S}_6$. The resulting simplified model is the single element cycle illustrated in Figure 7. Obviously, our analysis will apply more generally to systems with n service curve elements in a closed cycle for arbitrary n.

In the next subsection, we study the dynamics of the single element cycle illustrated in Figure 7.

B. Service Curves for a Closed Cycle

It follows from the definition of a throttle and by inspection of Figure 7 that $D = A \wedge T = A \wedge (B + W)$. Since $B \geq D * G$ and the window size $W(t)$ is bounded below by w^{min}, it thus follows that

$$D \geq A \wedge (D * G + w^{min}) \ . \tag{6}$$

Similarly, since $B \leq D * \bar{G}$ and $W(t) \leq w^{max}$, we obtain

$$D \leq A \wedge (D * \bar{G} + w^{max}). \tag{7}$$

We would like to obtain minimum and maximum service curve guarantees for the cycle. The main problem with the equations (6) and (7) which bound the departure process D from the throttle $\mathcal{T}$ is that they involve implicit inequalities. We would like to show that there exists solutions to the above inequalities and obtain tight lower and upper bounds on all such solutions D. We first focus on a lower bound to D.

To this end we first consider the inequality in (6) with an equality instead, i.e.

$$\tilde{D} = A \wedge (\tilde{D} * G + w^{min}) \ . \tag{8}$$

Clearly, any solution to (8) also satisfies (6).

Theorem 17: For any given causal arrival process A, there exists a unique departure process $\tilde{D}$ satisfying (8) and for any departure process D satisfying (6), $D \geq \tilde{D}$. Further $\tilde{D}$ may be obtained by the method of successive approximations, i.e. $\tilde{D} = \lim_{m\to\infty} \tilde{D}^m = \inf_{m\geq 0} \tilde{D}^m$, where $\tilde{D}^0 = A$, and

$$\tilde{D}^{m+1} = A \wedge (\tilde{D}^m * G + w^{min}) \qquad \text{for all } m \geq 0 \ . \tag{9}$$

Proof: See Appendix. □

Next, we use the result above to derive a minimum service curve for the throttle, i.e., represent the relationship between the exogenous and throttled arrival process in terms of a convolution operation. Toward this end, define $(G + w^{min})^{(m)}$ to be the m-fold convolution of $G + w^{min}$ with itself, e.g. $(G + w^{min})^{(1)} = G + w^{min}$, $(G + w^{min})^{(2)} = (G + w^{min}) * (G + w^{min})$, and so on. Also, define $(G + w^{min})^{(0)} = \delta$. Define

$$S_{\mathcal{T}} := \wedge_{m=0}^{\infty} (G + w^{min})^{(m)} \ . \tag{10}$$

Corollary 18: Any departure process D satisfying (6) also satisfies $D \geq A * S_{\mathcal{T}}$, i.e. the throttle has a minimum service curve of $S_{\mathcal{T}}$.

Proof: Define $S_{\mathcal{T}_m} := \wedge_{n=0}^{m} (G^{min} + w^{min})^{(n)}$. In view of Theorem 17, it suffices to show that $\tilde{D}^m, m \geq 0$ defined in the successive approximations procedure in Theorem 17 is given by

$$\tilde{D}^m = A * S_{\mathcal{T}_m} \ . \tag{11}$$

This is true for $m = 0$ by definition of $\tilde{D}^0$. Now assume it is true for some $m \geq 0$. Then

$$\begin{aligned}
\tilde{D}^{m+1} &= A \wedge (\tilde{D}^m * G + w^{min}) \\
&= A \wedge ((A * S_{\mathcal{T}_m}) * G + w^{min}) \\
&= (A * \delta) \wedge (A * S_{\mathcal{T}_m} * (G + w^{min})) \\
&= A * (\delta \wedge (S_{\mathcal{T}_m} * (G + w^{min}))) \\
&= A * S_{\mathcal{T}_{m+1}} \ .
\end{aligned}$$

□

Next, we consider a *maximum* service for the throttle. This can be obtained in a manner mirroring Theorem 17 and Corollary 18. However, as we will see, we may proceed somewhat more directly in this case. Toward this end, define $(\bar{G} + w^{max})^{(m)}$ to be the m-fold convolution of $\bar{G} + w^{max}$ with itself, and $(\bar{G} + w^{max})^{(0)} = \delta$. Define

$$\bar{S}_{\mathcal{T}} = \wedge_{m=0}^{\infty} (\bar{G} + w^{max})^{(m)} \ . \tag{12}$$

Proposition 19: Any departure process D satisfying (7) also satisfies $D \leq A * \bar{S}_{\mathcal{T}}$, i.e. the throttle has a maximum service curve of $\bar{S}_{\mathcal{T}}$.

Proof: Define $\bar{S}_{\mathcal{T}_m} = \wedge_{n=0}^{m} (\bar{G} + w^{max})^{(n)}$. It suffices to show for all $m \geq 0$ we have $D \leq A * \bar{S}_{\mathcal{T}_m}$, which we do by induction on m. For $m = 0$ we have $D \leq A * \delta = A$ from (7). Assume inductively then that $D \leq A * \bar{S}_{\mathcal{T}_m}$. ¿From (7) we have

$$\begin{aligned}
D &\leq A \wedge (D * \bar{G} + w^{max}) \\
&\leq A \wedge ((A * \bar{S}_{\mathcal{T}_m}) * \bar{G} + w^{max}) \\
&= (A \wedge \delta) \wedge (A * \bar{S}_{\mathcal{T}_m} * (\bar{G} + w^{max})) \\
&= A * (\delta \wedge (\bar{S}_{\mathcal{T}_m} * (\bar{G} + w^{max})) \\
&= A * \bar{S}_{\mathcal{T}_{m+1}} \ .
\end{aligned}$$

□

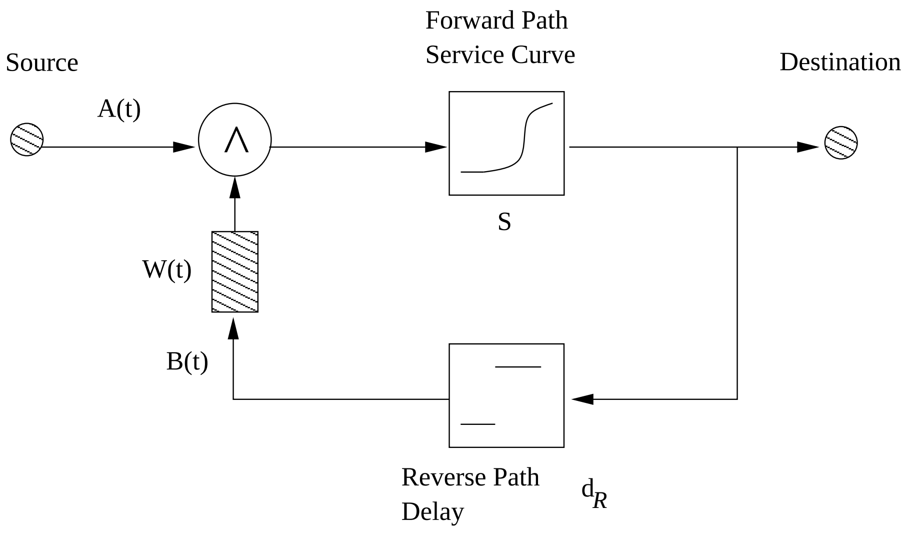

Fig. 8. A Collapsed Model of End-to-End Window Flow Control.

C. Discussion of End-to-End Window Flow Control

The results of the Section IV-B may be used to analyze end-to-end window flow control as discussed in Section IV-A. In particular, after a service curve is obtained for the throttle, an end-to-end service curve can be obtained by convolving the service curves of the elements in the forward path following the throttle with the service curve of the throttle.

In this subsection, we consider a somewhat simplified model for end-to-end flow control, illustrated in Figure 8, which is a special case of the model for end-to-end window flow control as presented in Section IV-A. For this model, we discuss how the end-to-end performance depends on the window size.

In the model illustrated in Figure 8, we have collapsed all the elements in the forward path between the throttle and the destination into a single service curve element, with corresponding minimum and maximum service curves S and $\bar{S}$. The reverse path which carries the acknowledgments back from the source to the throttle is modelled by a lumped system with delay bounded between d_R^{min} and d_R^{max}. Thus the reverse path is modelled by a service curve element with minimum and maximum service curves given by $\delta_{d_R^{max}}$ and $\delta_{d_R^{min}}$, respectively.

Let us now examine the minimum service curve of the system, say S_{sys}. The minimum service curve of the throttle $S_{\mathcal{T}}$ is given by Corollary 18, where we set $G = S * \delta_{d_R^{max}}$. Applying Proposition

10, the minimum service curve of the system is

$$\begin{aligned} S_{sys} &= S_{\mathcal{T}} * S \\ &= [\wedge_{m=0}^{\infty} (G + w^{min})^{(m)}] * S \\ &= \wedge_{m=0}^{\infty} [S * (G + w^{min})^{(m)}] \\ &= S \wedge [S * (G + w^{min})] \wedge [S * (G + w^{min})^{(2)}] \wedge \cdots \ . \end{aligned}$$

Now if the the minimum window size w^{min} is large enough so that $S * (G + w^{min}) \geq S$, then it follows from the above that $S_{sys} = S$, i.e. the minimum service curve of the system is the same as it would be if the throttle were not present. The condition that $S * (G + w^{min}) \geq S$ is equivalent to $w^{min} \geq S \oslash (S * G)(0)$, i.e.

$$\begin{aligned} w^{min} &\geq \sup_{t \in \mathbb{R}} \{S(t) - S * G(t)\} \\ &= \sup_{t \in \mathbb{R}} \{S(t) - S * S(t - d_R^{max})\} \\ &=: k^{min} \ . \end{aligned} \tag{13}$$

Thus, a minimum window size of k^{min} is sufficient to guarantee that the system minimum service curve is S, which would be the minimum service curve without the throttle. It is interesting to note that the equation (13) for k^{min} has the same form as the equation (4) for the the minimum amount of network buffering with an open loop strategy. In fact, the two quantities are identical when the maximum delay in the reverse path, d_R^{max}, is zero.

Mirroring the argument above, the maximum service curve of the system in Figure 8 is

$$\bar{S}_{sys} = \bar{S} \wedge [\bar{S} * (\bar{G} + w^{max})] \wedge [\bar{S} * (\bar{G} + w^{max})^{(2)}] \wedge \cdots \ .$$

Arguing as before, if $w^{max} \geq k^{max}$ then $\bar{S}_{sys} = \bar{S}$, where

$$k^{max} = \sup_{t \in \mathbb{R}} \{\bar{S}(t) - \bar{S} * \bar{S}(t - d_R^{min})\} \ . \tag{14}$$

For example, suppose the maximum rate that each router encountered by the session would serve the session is μ^{max} bits per second, and there is a total propagation delay of d_F^{min} in the forward direction. In this case, we may set the maximum service curve $\bar{S}$ to $\bar{S} = R_{\mu^{max}} * \delta_{d_F^{min}}$. This implies that a maximum window size of $k^{max} = \mu^{max}(d_R^{min} + d_F^{min})$ is necessary for the system to have a maximum service curve of at least $\bar{S} = R_{\mu^{max}} * \delta_{d_F^{min}}$. This is consistent with the well known intuition that a window size equal to the maximum bandwidth delay product is necessary to support the maximum bandwidth.

Suppose that a network provides a minimum service curve of S to a session. Let us now briefly compare the alternatives of open loop, rate based flow control, and closed loop, window based flow control. The source could either employ an open loop strategy whereby a regulator is used at the entry point of the network, or end-to-end window flow control strategy as above. In order to prevent buffer overflow with the open loop strategy, assuming a buffer allocation of b_{max}, the envelope E of the access regulator must satisfy $E(t) \leq S(t) + b_{max}$. Therefore the throughput of the access regulator cannot be greater than the asympotic slope of the minimum service curves of the routers. Thus, with the open loop strategy, the allocation of even very large buffers to the session will not increase the throughput. On the other hand, with window flow control, increasing the buffers allocated to the session allows a larger window size, which enables the possibility of supporting a greater throughput than the nominal amount specified by the minimum service curves.

If the network uses hop-by-hop flow control, the buffering requirements can be reduced somewhat. We discuss this next.

D. Hop-by-Hop Window Flow Control

Consider the system depicted in Figure 9, which models a network that employs hop-by-hop window flow control. In particular, at each hop a throttle is used, and the throttle is controlled by the departure process from the router at the next hop. The router at hop i is modelled by a service curve element with minimum and maximum service curves $S_{router,i}$ and $\bar{S}_{router,i}$. For simplicity of exposition[8], we assume that the departure process of the router at hop i encounters a fixed propagation delay d_{F_i} before reaching the throttle at hop $i+1$. Similarly, acknowledgments generated by the departure process of the router at hop $i+1$ encounter a fixed propagation delay of d_{R_i} before reaching the throttle at hop i. The throttle at hop i uses the window process W_i,

[8]More elaborate models can easily be handled.

where we assume that $0 < w_i^{min} \le W_i(t) \le w_i^{max}$. Note that the buffer requirement at router i is $\min\{W_{i-1}(t), W_i(t)\}$, $2 \le i \le n-1$, where n is the number of hops.

It is possible to analyze this system in the context of a general fork-join queueing model, and we refer the reader to [1] for details on this approach. Here, we analyze the system by exploiting the robustness of service curve definitions, and again by lumping network elements using Proposition 10. In particular, we shall iteratively use our analysis of the basic model in Figure 7.

Note that the throttle at hop i is contained in the cycle of elements which determine the throttle process at hop $i-1$. Thus, in order to determine a service curve for the throttle at hop $i-1$, we must first determine a service curve for the throttle at hop i. Since there is no throttle at the last hop, we may analyze the throttle at hop $n-1$ using the same method we used to analyze the simple cycle in Figure 7. Once a service curve for the throttle at hop $n-1$ is determined, we can incorporate that into the analysis of the throttle at hop $n-2$. By continuing in this manner, we may determine the service curves for each of the throttles, and then use Proposition 10 to determine the end-to-end service curve.

For example, suppose we let $S_i = S_{router,i} * S_{router,i+1} * \delta_{d_{F_i}}$ and $\bar{S}_i = \bar{S}_{router,i} * \bar{S}_{router,i+1} * \delta_{d_{F_i}}$. Parallel to equations (13) and (14), define

$$k_i^{min} = \sup_{t \in \mathbb{R}} \{ S_i(t) - S_i * S_i(t - d_{R_i}) \}$$

and

$$k_i^{max} = \sup_{t \in \mathbb{R}} \{ \bar{S}_i(t) - \bar{S}_i * \bar{S}_i(t - d_{R_i}) \} \ .$$

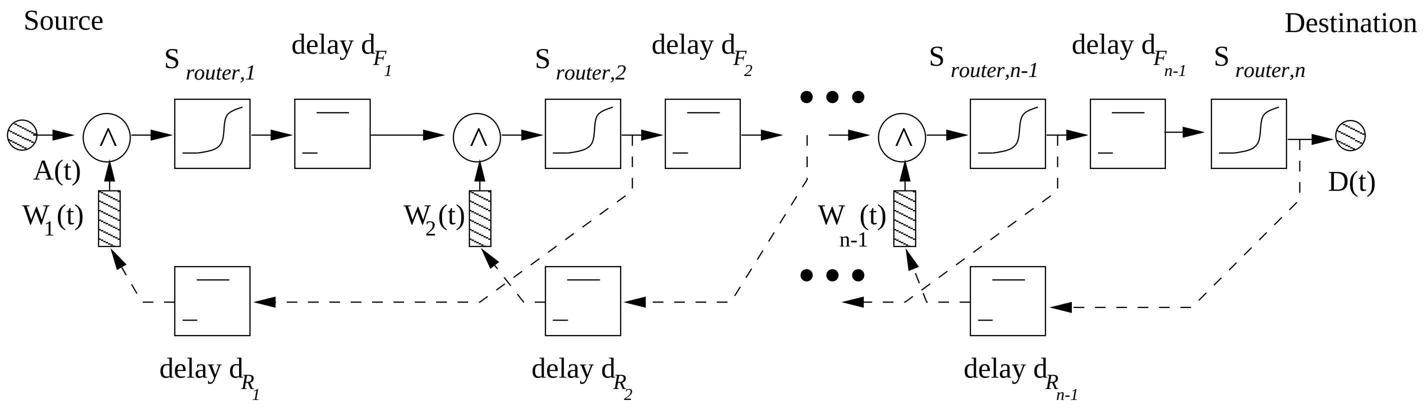

Fig. 9. An n-hop session with hop-by-hop window flow control.

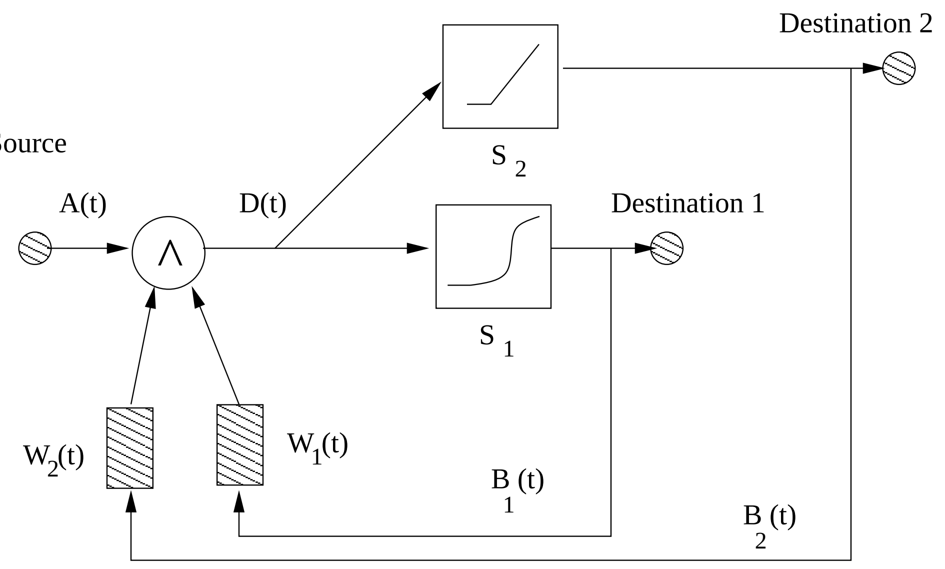

Fig. 10. A multicast session with window flow control.

By following the procedure outlined above, if for all $i < n$ and all t we have $k_i^{min} \leq W_i(t) \leq k_i^{max}$, it follows that the end to end service curves are identical to that obtained with no throttles present. In other words, in this case the end-to-end minimum and maximum service curves are $\delta_{d_F} * S_{router,1} * S_{router,2} * \cdots * S_{router,n}$ and $\delta_{d_F} * \bar{S}_{router,1} * \bar{S}_{router,2} * \cdots * \bar{S}_{router,n}$, where $d_F = \sum_{i=1}^{n-1} d_{F_i}$ is the total forward propagation delay. If we assume that each router supports a maximum service curve of $\bar{S}_{router,i} = R_{\mu^{max}}$, then we have $k_i^{max} = \mu^{max}(d_{F_i} + d_{R_i})$, which is the maximum bandwidth delay product over hop i. Thus, it is seen that in order to enable sessions to avail of excess bandwidth, smaller window sizes, and hence smaller buffer requirements, are required as compared to end-to-end window flow control. On the other hand, hop-by-hop flow control requires the network to actively participate in the flow control process by generating acknowledgments at each hop.

E. Window Flow Control with Multicast

As another application of the analysis of Section IV-B, consider the system illustrated in Figure 10, where data generated by a source is delivered to two destinations via distinct data paths. The data is passed through a throttle which is governed by feedback from each of the destinations. Specifically, the arrival process to the throttle is denoted by A and the departure process is denoted by D. The same departure process is fed to two network elements, 1 and 2. The i^{th} network element has departure process B_i, and guarantees minimum and maximum service curves S_i and $\bar{S}_i$. The throttle insures that at most W_i bits are stored in network element i. More specifically, the throttle operates such that

$$D = A \wedge (B_1 + W_1) \wedge (B_2 + W_2) \ .$$

Assume that the window sizes are bounded according to $w_i^{min} \leq W_i(t) \leq w_i^{max}$, where $w_i^{min} > 0$.

Using the definition of a minimum service curve and the distributive property of convolution we have

$$\begin{aligned} D &\geq A \wedge (D * S_1 + w_1^{min}) \wedge (D * S_2 + w_2^{min}) \\ &= A \wedge (D * (S_1 + w_1^{min})) \wedge (D * (S_2 + w_2^{min})) \\ &= A \wedge (D * [(S_1 + w_1^{min}) \wedge (S_2 + w_2^{min})]) \\ &= A \wedge (D * (G + w^{min})) \\ &= A \wedge (D * G + w^{min}) \ , \end{aligned}$$

where $w^{min} = \min\{w_1^{min}, w_2^{min}\}$ and $G = [(S_1 + w_1^{min}) \wedge (S_2 + w_2^{min})] - w^{min}$. It is easy to verify that G is a causal process, so that we obtain exactly the same form as in (6). The minimum service curve for the throttle can then be derived as before. Similarly, using the definition of a maximum service curve and the distributive property of convolution, we have

$$D \leq A \wedge (D * \bar{G} + w^{max}) \ ,$$

where $\bar{G} = [(\bar{S}_1 + w_1^{max}) \wedge (\bar{S}_2 + w_2^{max})] - w^{max}$, and $w^{max} = \min\{w_1^{max}, w_2^{max}\}$. Hence the maximum service curve of the throttle can be obtained as before.

Obviously, this generalizes easily to more than two destinations. Furthermore, it is possible to analyze a large class of complex configurations of network elements by lumping together network elements as in the previous subsections, using the analysis of the basic feedback model in Figure 7.

V. Conclusions

In this paper, we have presented a new mathematical model, the service curve element, to describe the operation of a variety of network elements. We have obtained performance bounds for a flow traversing a service curve element. While some of these bounds are not new, the proofs here are more elegant in using a simple calculus based on the convolution operation. It is apparent that the framework is related to the theory of linear filtering, where the concept of a service curve here is somewhat analogous to the impulse response of a linear system. This analogy is discussed at more length in [10].

We have analyzed unicast guaranteed sessions as a series of service curve elements, and obtained bounds on end-to-end delay and buffering requirements. This paper also suggested how regulation may be used with the network to reduce resource requirements for a session without compromising negotiated performance. We then considered the adaptive service, which uses window flow control. This service is appropriate for sessions that may wish to send larger amounts of traffic into the network when resources are lightly loaded, while reserving a certain amount of bandwidth at all times. In modelling such a session, we used a cycle of service curve elements with a throttle to restrict the traffic arriving into the network. We derived novel performance bounds for such a session (unicast or multicast), and used these to consider tradeoffs between window size and the performance of the adaptive session. Finally, we considered hop-by-hop windows as a means of further reducing buffering requirements within the network.

Appendix

Proof of Lemma 3. We have

$$\begin{aligned} [(A * (B \oslash C)] * C &= A * [(B \oslash C) * C] \\ &\geq A * B \ . \end{aligned}$$

Since the smallest function H satisfying $H * C \geq A * B$ is $(A * B) \oslash C$, it thus follows from the above that $A * (B \oslash C) \geq (A * B) \oslash C$. □

Proof of Lemma 4. By Lemma 3, we have $(B \oslash B) * (B \oslash B) \geq [(B \oslash B) * B] \oslash B \geq B \oslash B$. □

Proof of Proposition 15. The end-to-end minimum service curve of the system with a regulator inserted before any of the elements is $S * E$. Thus, if D is the departure process from the entire system, corresponding to the arrival process A, we have

$$D \geq A * (S * E)$$

$$
\begin{aligned}
&= A * S * [(B * S) \oslash (B * S)] \\
&= (A * B) * S * [(B * S) \oslash (B * S)] \\
&= A * (B * S) * [(B * S) \oslash (B * S)] \\
&\geq A * B * S \\
&= A * S \; .
\end{aligned}
$$

Thus, the end-to-end minimum service curve is unchanged with the insertion of a regulator before any of the elements. □

Proof of Theorem 17. Let $\{D^\alpha, \alpha \in \mathcal{A}\}$ be the set of causal processes satisfying the inequality

$$D^\alpha \geq A \wedge (D^\alpha * G + w^{min}). \tag{15}$$

Clearly, the set is non-empty as $D = A$ satisfies (15). Next, it is easy to verify that $\tilde{D} := \inf_{\alpha \in \mathcal{A}} D^\alpha$ is a causal process that also satisfies (15). We claim that in fact $\tilde{D}$ satisfies (8). Assume not. Define $\tilde{D}'$ to be

$$\tilde{D}' := A \wedge (\tilde{D} * G + w^{min}) \; . \tag{16}$$

First, it is easily verified that $\tilde{D}'$ is a causal process. Also, since $\tilde{D}$ satisfies (15) we have $\tilde{D}' \leq \tilde{D}$. Thus, replacing $\tilde{D}$ with $\tilde{D}'$ in (16), we have

$$\tilde{D}' \geq \; A \wedge (\tilde{D}' * G + w^{min}) \; .$$

Thus, it follows that $\tilde{D}'$ also satisfies (15). By assumption, however, $\tilde{D}$ does not satisfy (8), and hence $\tilde{D}'(t) < \tilde{D}(t)$ for at least one value of t. This contradicts the minimality of $\tilde{D}$ as a solution to (15) and establishes that $\tilde{D}$ satisfies (8).

Next, we establish that there is a unique causal process D that satisfies (8). Assume not. Let $\tilde{D}$ and $\tilde{D}'$ be two different processes satisfying (8). Let

$$t_0 := \inf\{t \in \mathbb{R} : \tilde{D}(t) \neq \tilde{D}'(t)\}.$$

Note that $t_0 \geq 0$ as $\tilde{D}(t) = \tilde{D}'(t) = 0$ for all $t < 0$. By the right continuity of $\tilde{D}$ and $\tilde{D}'$, there exists $\alpha > 0$ such that for all $t \in [t_0, t_0 + \alpha)$, $\tilde{D}(t) < \tilde{D}(t_0) + w^{min}$ and $\tilde{D}'(t) < \tilde{D}'(t_0) + w^{min}$. We now show that $\tilde{D}(t) = \tilde{D}'(t)$ for all $t \in [t_0, t_0 + \alpha)$, which contradicts the definition of t_0, and hence establishes uniqueness. For $t \in [t_0, t_0 + \alpha)$ we have

$$\begin{aligned}
\tilde{D}(t) &= A(t) \wedge (\tilde{D} * G(t) + w^{min}) \\
&= A(t) \wedge \inf_{\tau < t_0} [\tilde{D}(\tau) + G(t-\tau) + w^{min}] \wedge \inf_{\tau \geq t_0} [\tilde{D}(\tau) + G(t-\tau) + w^{min}] \\
&= A(t) \wedge \inf_{\tau < t_0} [\tilde{D}(\tau) + G(t-\tau) + w^{min}] \\
&= A(t) \wedge \inf_{\tau < t_0} [\tilde{D}'(\tau) + G(t-\tau) + w^{min}] \\
&= \tilde{D}'(t) \ ,
\end{aligned}$$

where the third equality above holds because $\inf_{\tau \geq t_0}[\tilde{D}(\tau) + G(t-\tau) + w^{min}] \geq \tilde{D}(t_0) + w^{min} > \tilde{D}(t)$, the fourth because $\tilde{D}(\tau) = \tilde{D}'(\tau)$ for $\tau < t_0$, and the last by interchanging $\tilde{D}$ and $\tilde{D}'$ in the preceeding steps.

Finally, we show that $\tilde{D}^m, m \geq 0$ defined in the theorem converge to this unique solution. It is easy to establish by induction that $\tilde{D}^m, m \geq 0$ is a non-increasing sequence and hence has a limit $\tilde{D}^\infty := \lim_{m\to\infty} \tilde{D}^m = \inf_{m \geq 0} \tilde{D}^m$. Thus,

$$\begin{aligned}
\tilde{D}^\infty &= \inf_{m \geq 0} \tilde{D}^m = \inf_{m \geq 0} \tilde{D}^{m+1} \\
&= \inf_{m \geq 0} [A \wedge (\tilde{D}^m * G + w^{min})] \\
&= A \wedge ([\inf_{m \geq 0} \tilde{D}^m] * G + W_{min}) \\
&= A \wedge (\tilde{D}^\infty * G + w^{min}).
\end{aligned}$$

□

References

[1] R. Agrawal and R. Rajan. Performance bounds for guaranteed and adaptive services. Technical Report RC 20649, IBM Research Division, 1996.

[2] J. C. R. Bennett and H. Zhang. Hierarchical packet fair queueing algorithms. In *Proc. ACM SIGCOMM'96 Conference*, pages 143–156, Stanford University, CA, August 1996.

[3] C.-S. Chang. Stability, queue length, and delay of deterministic and stochastic queueing networks. *IEEE Trans. on Automatic Control,* 39:913-931, May, 1994.

[4] C.-S. Chang. On deterministic traffic regulation and service guarantee: a systematic approach by filtering. Proceedings of IEEE INFOCOM'97.

[5] R. L. Cruz. A calculus for network delay, part I: Network elements in isolation. *IEEE Trans. Inform. Theory*, 37:114–131, 1991.

[6] R. L. Cruz. A calculus for network delay, part II: Network analysis. *IEEE Trans. Inform. Theory*, 37:132–141, 1991.

[7] R. L. Cruz. Service burstiness and dynamic burstiness measures: a framework. *Journal of High Speed Networks*, vol. 1, no. 2, pp. 105-127, 1992.

[8] R. L. Cruz. Quality of service guarantees in virtual circuit switched networks. *IEEE Journal on Selected Areas in Communications*, 13:1048–1056, August 1995.

[9] R. L. Cruz. SCED+: Efficient management of quality of service guarantees. To appear in *Proc. INFOCOM'98*, IEEE Computer Society Press, March, 1998.

[10] R. L. Cruz and C. M. Okino. Service guarantees for window flow control. *Proceedings of the 34th Allerton Conference on Communication, Control, & Computing* , Monticello, IL, October, 1996.

[11] L. Georgiadis, R. Guérin, and A. Parekh. Optimal multiplexing on a single link: delay and buffer requirements. *Proc. IEEE INFOCOM'94,* vol. 2, 1994, pp. 524-532 also to appear in *IEEE Transactions on Information Theory.*

[12] L. Georgiadis, R. Guerin, V. Peris, and R. Rajan. Efficient support of delay and rate guarantees in an internet. In *Proc. ACM SIGCOMM'96 Conference*, pages 106–116, Stanford University, CA, August 1996.

[13] S.J. Golestani. Congestion-free communication in high-speed packet networks. *IEEE Trans. on Communications,* vol. 39, no. 12, Dec. 1991, pp. 1802-1812.

[14] S. J. Golestani. A self-clocked fair queueing scheme for broadband applications. In *Proceedings of IEEE INFOCOM'94*, pages 636–646, Toronto, Canada, June 1994.

[15] P. Goyal, S. S. Lam, and H. M. Vin. Determining end-to-end delay bounds in heterogeneous networks. *ACM/Springer-Verlag Multimedia Systems Journal*, 1996. To appear.

[16] A. Hung and G. Kesidis. Bandwidth scheduling for wide-area ATM networks using virtual finishing times. *IEEE/ACM Transactions on Networking,* vol. 4., no. 1, pp . 49-54, Feb. 1996.

[17] T. Konstantopoulos and V. Anantharam. Optimal flow control schemes that regulate the burstiness of traffic. *IEEE/ACM Transactions on Networking*, 3(4):450–458, August 1995.

[18] J.-Y. Le Boudec. Network calculus made easy. preprint, 1996.

[19] J. Liebeherr, D.E. Wrege, and D. Ferrari. Exact admission control for networks with a bounded delay service. *IEEE/ACM Transactions on Networking,*, vol. 4 no. 6, December 1996, pp. 885-901.

[20] A. K. Parekh. *A Generalized Processor Sharing Approach to Flow Control in Integrated Services Networks.* PhD thesis, Laboratory for Information and Decision Systems, Massachusetts Institute of Technology, Cambridge, MA 02139, February 1992. No. LIDS-TH-2089.

[21] A. K. Parekh and R. G. Gallager. A generalized processor sharing approach to flow control in integrated services networks: The single node case. *IEEE/ACM Transactions on Networking*, 1(3):344–357, June 1993.

[22] A. K. Parekh and R. G. Gallager. A generalized processor sharing approach to flow control in integrated services networks: The multiple node case. *IEEE/ACM Transactions on Networking*, 2(2):137–150, April 1994.

[23] S. Shenker and C. Partridge. Specification of guaranteed quality of service. Internet Draft draft-ietf-intserv-guaranteed-svc-03.txt, Nov 1995.

[24] H. Sariowan, R. L. Cruz, and G. C. Polyzos. Scheduling for quality of service guarantees via service cruves. *Proc. International Conference on Computer Communications and Networks '95 (ICCCN'95)*, Sept. 1995, pp. 512-20.

[25] H. Sariowan. A service-curve approach to performance guarantees in integrated-service networks. *Ph.D. Dissertation in Electrical and Computer Engineering*, University of California, San Diego, June 1996.

[26] D. Stiliadis and A. Varma. Latency-rate servers: A general model for analysis of traffic scheduling algorithms. In *Proc. IEEE INFOCOM'96*, pages 111–119, San Francisco, March 1996.

[27] I. Stoica, H. Zhang, and T.S. Eugene Ng. A hierarchical fair service curve algorithm for link-sharing, real-time, and priority services. *Proceedings of the 1997 ACM SIGCOMM Conference.*

[28] B. Suter, T. V. Lakshman, D. Stiliadis, and A. K. Choudhury. Design considerations for supporting TCP with per-flow queueing. preprint, 1998.

[29] J. Turner. New directions in communications (or Which way to the information age?). *IEEE Communications Magazine*, Vol 24., No 10, October 1986.

[30] J. Wroclawski. Specification of the controlled-load network element service. Internet Draft draft-ietf-intserv-ctrl-load-svc-01.txt, Nov 1995.

[31] H. Zhang and D. Ferrari. Rate-controlled static priority queueing. In *Proceedings of IEEE INFOCOM'93*, pages 227–236, San Francisco, April 1993.

[32] H. Zhang and D. Ferrari. Rate-controlled service disciplines. *Journal of High Speed Networks*, 3(4):389–412, 1994.

[33] L. Zhang. Virtual clock: A new traffic control algorithm for packet switching networks. In *Proceedings of the ACM SIGCOMM'90*, pages 19–29, Philadelphia, PA, Sept. 1990.